DNN Speech Recognizer

REVIEW

CODE REVIEW

HISTORY

Meets Specifications

This is a strong submission, great work!
Congratulation on passing the project! 🥰💯🎉

FURTHER READING IN ASR:

- Guide to ASR
- ASR is devices
- Research Paper which covers different ASR models

STEP 2: Model 0: RNN

✓

The submission trained the model for at least 20 epochs, and none of the loss values in `model_0.pickle` are undefined. The trained weights for the model specified in `simple_rnn_model` are stored in `model_0.h5`.

The submission trained the model for 20 epochs, and all of the loss values are defined and saved in the `model_0.pickle`.

The trained weights for the model specified in `simple_rnn_model` are stored in `model_0.h5`.

STEP 2: Model 1: RNN + TimeDistributed Dense

✓

The submission includes a `sample_models.py` file with a completed `rnn_model` module containing the correct architecture.

The `sample_models.py` contains the completed `rnn_model` module - containing the correct architecture.

✓

The submission trained the model for at least 20 epochs, and none of the loss values in **`model_1.pickle`** are undefined. The trained weights for the model specified in **`rnn_model`** are stored in **`model_1.h5`**.

The submission trained the model for 20 epochs, and all of the loss values are defined and saved in the `model_1.pickle`.

The trained weights for the model specified in `rnn_model` are stored in `model_1.h5`.

STEP 2: Model 2: CNN + RNN + TimeDistributed Dense

✓

The submission includes a **`sample_models.py`** file with a completed **`cnn_rnn_model`** module containing the correct architecture.

The `sample_models.py` contains the completed `cnn_rnn_model` module - containing the correct architecture.

✓

The submission trained the model for at least 20 epochs, and none of the loss values in **`model_2.pickle`** are undefined. The trained weights for the model specified in **`cnn_rnn_model`** are stored in **`model_2.h5`**.

The submission trained the model for 20 epochs, and all of the loss values are defined and saved in the `model_2.pickle`.

The trained weights for the model specified in `cnn_rnn_model` are stored in `model_2.h5`.

STEP 2: Model 3: Deeper RNN + TimeDistributed Dense

✓

The submission includes a **`sample_models.py`** file with a completed **`deep_rnn_model`** module containing the correct architecture.

The `sample_models.py` contains the completed `deep_rnn_model` module - containing the correct architecture.

✓

The submission trained the model for at least 20 epochs, and none of the loss values in **`model_3.pickle`** are undefined. The trained weights for the model specified in **`deep_rnn_model`** are stored in **`model_3.h5`**.

The submission trained the model for 20 epochs, and all of the loss values are defined and saved in the `model_3.pickle`.

The trained weights for the model specified in `deep_rnn_model` are stored in `model_3.h5`.

STEP 2: Model 4: Bidirectional RNN + TimeDistributed Dense

✓

The submission includes a **`sample_models.py`** file with a completed **`bidirectional_rnn_model`** module containing the correct architecture.

The `sample_models.py` contains the completed `bidirectional_rnn_model` module - containing the correct architecture.

✓

The submission trained the model for at least 20 epochs, and none of the loss values in `model_4.pickle` are undefined. The trained weights for the model specified in `bidirectional_rnn_model` are stored in `model_4.h5`.

The submission trained the model for 20 epochs, and all of the loss values are defined and saved in the `model_4.pickle`.

The trained weights for the model specified in `bidirectional_rnn_model` are stored in `model_4.h5`.

STEP 2: Compare the Models

✓

The submission includes a detailed analysis of why different models might perform better than others.

The submission includes the detailed analysis of why different models might perform better than others.

Suggestion: You could plot an additional graph of all models, excluding the `model_0` - so as to better analyze the rest of them.

STEP 2: Final Model

✓

The submission trained the model for at least 20 epochs, and none of the loss values in `model_end.pickle` are undefined. The trained weights for the model specified in `final_model` are stored in `model_end.h5`.

The submission trained the model for 20 epochs, and all of the loss values are defined and saved in the `model_end.pickle`.

The trained weights for the model specified in `final_model` are stored in `model_end.h5`.

✓

The submission includes a `sample_models.py` file with a completed `final_model` module containing a final architecture that is not identical to any of the previous architectures.

The `sample_models.py` contains the completed `final_model` module - containing the correct architecture.

Suggestion: You may consider using *data augmentation*, so as to improve the performance further of your `final_model`.

✓

The submission includes a detailed description of how the final model architecture was designed.

Nice job!

The submission includes a detailed description of how the final model architecture was designed.
Your reasoning is logical here.

Further question to ponder upon - Did the `final_model` performed as per your expectation? Why or why not?

DOWNLOAD PROJECT

RETURN TO PATH

Rate this review

START